KUL: Recognition and Normalization of Temporal Expressions

Oleksandr Kolomiyets, Marie-Francine Moens
Department of Computer Science
Katholieke Universiteit Leuven
{oleksandr.kolomiyets, sien.moens}@cs.kuleuven.be

Abstract

In this paper we describe a system for the recognition and normalization of temporal expressions (Task 13: TempEval-2, Task A). The recognition task is approached as a classification problem of sentence constituents and the normalization is implemented in a rule-based manner. One of the system features is extending positive annotations in the corpus by semantically similar words automatically obtained from a large unannotated textual corpus. The best results obtained by the system are 0.85 and 0.84 for precision and recall respectively for recognition of temporal expressions; the accuracy values of 0.91 and 0.55 were obtained for the feature values `TYPE` and `VAL` respectively.

1 Introduction

Recognition of temporal expressions[1] is a task of proper identification of phrases with temporal semantics in running text. After several evaluation campaigns targeted at temporal processing of text, such as MUC, ACE TERN and TempEval-1 (Verhagen et al., 2007), the recognition and normalization task has been again newly reintroduced in TempEval-2 (Pustejovsky & Verhagen, 2009). The task is defined as follows: determine the extent of the time expressions; in addition, determine the value of the features `TYPE` for the type of the temporal expression and its temporal value `VAL`. In this paper we describe the KUL system that has participated in this task.

Architecturally, the system employs a pipelined information processing chain and implements a number of machine learning classifiers for extracting the necessary information for the temporal value estimation. The normalization step employs a number of hand-crafted vocabularies for tagging single elements of a temporal expression and a rule-based system for estimating the temporal value. The performance of the system obtained the values of 0.85 and 0.84 for precision and recall respectively for the recognition of temporal expressions. The accuracy for the type and value is 0.91 and 0.55 respectively.

The remainder of the paper is organized as follows: Section 2 reports on the architecture of the system with single modules and describes theirs functions. Section 3 presents the results and error analysis; the conclusions are provided in Section 4.

2 System Architecture

The system is implemented in Java and follows a pipelined method for information processing. Regarding the problems it solves, it can be split in two sub-systems: recognition and normalization.

2.1 Recognition of Temporal Expressions

This sub-system is employed for finding temporal expressions in the text. It takes a sentence as input and looks for temporal expressions in it.

Pre-processing: At this step the input text undergoes syntactic analysis. Sentence detection, tokenization, part-of-speech tagging and parsing are applied[2].

Candidate selection: Since only certain lexical categories can be temporal expressions and they are defined in the TIDES standard (Ferro et

[1] Temporal expressions are sometimes referenced as time expressions and timexes.

[2] For preprocessing we use the OpenNLP package (http://opennlp.sourceforge.net).

al., 2003), in our implementation we consider the following chunk-phrases as candidates for temporal expressions: nouns (*week*, *day*), proper names (*Tuesday*, *May*), noun phrases (*last Tuesday*), adjectives (*current*), adjective phrases (*then current*), adverbs (*currently*), adverbial phrases (*a year ago*), and numbers (*2000*). As input it takes the sentences with provided syntactic information and marks phrases in the parse tree belonging to the above types for temporal expressions.

Annotation alignment: If the system is used for training classifiers, all the candidates in a sentence are examined against the available annotations. The candidates, whose parse and annotation extents aligned, are taken as positive examples and the rest is considered as negative.

Feature Design: To produce a feature-vector we use most valuable features extracted for phrase-candidate. After a number of experiments the following features were selected:

- Last token in the phrase, most probable token to be a temporal trigger;
- Lemma of the last phrasal token;
- Part-of-speech of the last phrasal token;
- Character pattern of the last phrasal token as introduced in (Ahn et al., 2007);
- Neighbor POS's. The concatenated part-of-speech tags of the last phrasal token and its preceding token;
- Character pattern of the entire phrase;
- Phrase surface. A concatenated string of sub-parse types for the phrase;
- A Boolean feature indicating nested complex phrasal parses, such as noun verb, adverbial, adjective or prepositional phrase;
- Depth of the phrase. The number of the nested sub-parses to the deepest pre-terminal sub-parse.

All the features are considered as Boolean.

Classification: Once the classifiers are trained they can be used for recognition of temporal expressions on test sentences. A preprocessed sentence is taken as input and starting from its parse-tree root the candidate-phrases are classified. The most probable class will be assigned to the candidate under consideration. Once the phrase is classified as temporal expression no further classification of nested phrases is performed, since no embedded timexes are allowed in the corpus. After a series of experiments with different machine learning techniques on the training data the maximum entropy classifier was chosen.

Extending positive instances: Sparseness of annotated corpora is the biggest challenge for any supervised machine learning technique. To overcome this problem we hypothesize that knowledge of semantic similar words could be found by associating words that do not occur in the training set to similar words that did occur in the training set. Furthermore, we would like to learn these similarities automatically in order to be as much as possible independent of knowledge sources that might not be available for all languages or domains. For example, there is in TimeBank a temporal expression "*last summer*" with the temporal trigger *summer*, but there is no annotation of temporal expressions built around the temporal trigger *winter*, and this means that no temporal expression with the trigger *winter* can be recognized. Something similar usually happens to any annotated corpus and we want to find a way how to find other temporal expressions outside the available data, which can be used for training. On the other hand, we want to avoid a naïve selection of words as, for example, from a gazetteer with temporal triggers, which may contradict with grammatical rules and the lexical context of a timex in text, e.g.:

on *Tuesday* said....

But grammatically wrong by naïve replacement from a gazetteer:

… on ***week*** said*…
… on ***day*** said*…
… on ***month*** said* …

In order to find these words, which are legitimate at a certain position in a certain context we use the latent word language model (LWLM) (Deschacht & Moens, 2009) with a Hidden Markov Model approach for estimating the latent word parameters.

Complementary, we use WordNet (Miller, 1995) as a source that can provide a most complete set of words similar to the given one. One should note that the use of WordNet is not straight-forward. Due to the polysemy, the word sense disambiguation (WSD) problem has to be solved. Our system uses latent words obtained by the LWLM and chooses the synset with the high-

est overlap between WordNet synonyms and coordinate terms, and the latent words. The overlap value is calculated as the sum of LWLM probabilities for matching words.

Having these two sets of synonyms and after a series of preliminary tests we found the setting, at which the system produces the highest results and submitted several runs with different strategies:

- Baseline (no expansion) (KUL Run 1)
- 3 LWLM words with highest probabilities (KUL Run 2)
- 3 WordNet coordinate terms; WSD is solved by means of LWLM[3] (KUL Run 3)

For each available annotation in the corpus a positive instance is generated. After that, the token at the most probable position for a temporal trigger is replaced by a synonym from the synonym set found to the available token.

2.2 Normalization of Temporal Expressions

Normalization of temporal expressions is a process of estimating standardized temporal values and types. For example, the temporal expression "*summer 1990*" has to be resolved to its value of `1990-SU` and the type of `DATE`. In contrast, for the expression "*last year"* the value cannot be estimated directly, rather it gets a modified value of another time expression.

Due to a large variance of expressions denoting the same date and vagueness in language, rule-based systems have been proven to perform better than machine-learning ones for the normalization task. The current implementation follows a rule-based approach and takes a pre-processed document with recognized temporal expressions (as it is described in Section 2.1) and estimates a standardized ISO-based date/time value. In the following sections we provide implementation details of the system.

Before the temporal value is estimated, we employ a classifier, which uses the same feature sets and classify the temporal expression among type classes `DATE`, `TIME`, `DURATION` and `SET`.

Labeling: Labeling text is a process of providing tags to tokens of chunk-phrases from a defined set of tags. We carefully examined available annotated temporal expressions and annotation standards to determine categories of words participating in temporal expressions. The following set of categories with labels based on semantics of temporally relevant information and simple syntax was defined: ordinal numbers (*first*, *30th* etc.), cardinal numbers (*one*, *two*, *10* etc.), month names (*Jan.*, *January* etc.), week day names (*Mo.*, *Monday* etc.), season names (*summer*, *winter* etc.), parts of day (*morning*, *afternoon* etc.), temporal directions (*ago*, *later*, *earlier* etc.), quantifiers (*several*, *few* etc.), modifiers (*recent*, *last* etc.), approximators (*almost*, *nearly* etc.), temporal co-references (*time*, *period* etc.), fixed single token timexes (*tomorrow*, *today* etc.), holidays (*Christmas*, *Easter* etc.) and temporal units (*days*, *months*, *years* etc.). Also fine-grained categories are introduced: day number, month number and year number. For each category we manually construct a vocabulary, in which each entry specifies a value of a temporal field or a final date/time value, or a method with parameters to apply.

As input, the normalization takes a recognized temporal expression and its properties, such as the temporal type and the discourse type[4]. During labeling each token in a temporal expression is tagged with one or multiple labels corresponding to the categories defined above. For each of the categories a custom detector is implemented. The detector declares the method to run and the expected type of the result. The rules that implement the logics for the detector are inherited from an abstract class for this specific detector, so that if a new rule needs to be implemented its realization is limited to the development of one class, all the rest the detector does automatically. Besides, the order, in which detectors have to be run, can be specified (as for example, in case of fine-grained detectors). As output, the module provides labels of the categories to the tokens in the temporal expression. If there is no entry in the vocabulary for a token, its part-of-speech tag is used as the label.

Value estimation: Value estimation is implemented in the way of aggregating the values defined for entries in the vocabulary and/or executing instructions or methods specified. Also a set of predefined resolution

[3] Preliminary experiments, when the most common sense in WordNet is chosen for increasing the number of positive examples, showed a low performance level and thus has not been proposed for evaluations.

[4] Since in TempEval-2 the reference to the timex with respect to which the value estimated is given, the normalization module considers all timexes as deictic.

rules is provided and can be extended with new implementations of resolution strategies.

For resolution of complex relative temporal expressions, the value for which cannot be estimated directly, we need to rely on additional information found at the recognition step. This includes the semantic type of the timex, discourse type and contextual temporal information (speech or document creation time, or previously mentioned timexes). Let's consider the following temporal expression as an example: *10 days ago*. In this example the temporal expression receives a modified value of another timex, namely the value of the document creation time. The temporal expression is recognized and classified as a date (`SEM TYPE: DATE`), which refers to another timex (`DISCOURSE TYPE: DEICTIC`). It takes the value of the referenced timex and modifies it with respect to the number (*10*), magnitude (*days*) and temporal direction (*ago*). Thus, the final value is calculated by subtracting a number of days for the value of the referenced timex.

3 Results and Error Analysis

In the Table 1 the results of the best-performing runs are presented.

Run	Recognition			Normalization	
	P	*R*	*F1*	`TYPE` Acc.	`VAL` Acc.
1	0.78	0.82	0.8	0.91	0.55
2	0.75	0.85	0.797	0.91	0.51
3	0.85	0.84	0.845	0.91	0.55

Table 1. Results of different runs of the system.

As we can see the best results were obtained by extending available annotations with maximum 3 additional instances, which are extracted as coordinate terms in WordNet, whereas the WSD problem was solved as the greatest overlap between coordinate terms and latent words obtained by the LWLM.

Most of the errors at the recognition step were caused by misaligned parses and annotations.

For normalization we acknowledge the significance of estimating a proper temporal value with a correct link to the temporal expression with its value. In the TempEval-2 training data the links to the temporal expressions indicating how the value is calculated were not provided, and thus, the use of machine learning tools for training and automatic disambiguation was not possible. We choose a fixed strategy and all relative temporal expressions were resolved with respect to the document creation time, which caused errors with wrong temporal values and a low performance level.

4 Conclusions

For TempEval-2 we proposed a system for the recognition and normalization of temporal expressions. Multiple runs were submitted, among which the best results were obtained with automatically expanded positive instances by words derived as coordinate terms from WordNet for which the proper sense was found as the greatest overlap between coordinate terms and latent words found by the LWLM.

Acknowledgements

This work has been funded by the Flemish government as a part of the project AMASS++ (Grant: IWT-60051) and by Space Applications Services NV as part of the ITEA2 project LINDO (ITEA2-06011, IWT-70043).

References

Ahn, D., van Rantwijk, J., and de Rijke, M. 2007. A Cascaded Machine Learning Approach to Interpreting Temporal Expressions. In *Proceedings of NAACL-HLT 2007*.

Deschacht, K., and Moens M.-F. 2009. Using the Latent Words Language Model for Semi-Supervised Semantic Role Labeling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Ferro, L., Gerber, L., Mani, I., Sundheim, B., and Wilson, G. 2003. TIDES 2003 Standard for the Annotation of Temporal Expressions.

Miller, G. A. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11): 39-41.

Pustejovsky, J. and Verhagen, M. 2009. SemEval-2010 Task 13: Evaluating Events, Time Expressions, and Temporal Relations (TempEval-2). In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*.

Verhagen, M., Gaizauskas, R., Schilder, F., Hepple, M., and Pustejovsky, J. 2007. Semeval-2007 Task 15: Tempeval Temporal Relation Identification. In *SemEval-2007: 4th International Workshop on Semantic Evaluations*.